实验报告

cache设计实验

姓名： 陈锦赐

学号： 141220008

14级计算机科学与技术系 1班

邮箱： njucjc@163.com

时间：2017年1月日星期日

**一、实验目的**

1.了解cache的内部构造

2.掌握cache控制器的原理及其设计方法

3.了解cache替换算法

4.学会设计有限状态机

**二、实验环境**

1.装有Vivado的计算机一台

2.Nexy4开发板

**三、实验原理**

1.Cache的对外接口

(1)处理器-Cache-DDR之间的接口图



(2)信号含义说明表

|  |  |  |  |
| --- | --- | --- | --- |
| 信号 | 方向 | 宽度 | 信号说明 |
| instr\_req | core->cache | 1 | 来自core的读指令请求 |
| data\_r\_req | core->cache | 1 | 来自core的读数据请求 |
| data\_w\_req | core->cache | 1 | 来自core的写数据请求 |
| byte\_w\_en | core->cache | 4 | 来自core的数据字节写使能 |
| instr\_addr | core->cache | 32 | 来自core的指令地址 |
| data\_addr | core->cache | 32 | 来自core的读/写数据地址 |
| data\_in | core->cache | 32 | 来自core的写入数据 |
| ddr\_response | ddr->cache | 1 | 来自DDR的访存反馈 |
| line\_in | ddr->cache | 255 | ddr载入cache的数据块 |
| mem\_stall | cache->core | 1 | 标识当前还有来自core的请求未完成 |
| data\_out | cache->core | 32 | 从cache读取的数据 |
| instr\_out | cache->core | 32 | 从cache读取的指令 |
| ram\_en\_out | cache->ddr | 1 | Cache发送给DDR的信号，表示要读写DDR |
| ram\_write\_out | cache->ddr | 1 | 1表示写DDR，0表示读DDR |

1. cache顶层模块

(1)顶层模块图



注：上图主要分为三个模块：cache\_control、i\_cache、d\_cache分别表示cache控制器、指令cache、数据cache

(2)cache控制器

i)控制器接口图



ii)控制器接口信号说明

|  |  |  |  |
| --- | --- | --- | --- |
| 信号名 | 类型 | 宽度 | 信号说明 |
| instr\_req | input | 1 | 读指令请求 |
| data\_r\_req | input | 1 | 读数据请求 |
| data\_w\_req | input | 1 | 写数据请求 |
| hit | input | 2 | cache读写是否命中(i/d\_cache) |
| dirty | input | 2 | cache行是否为脏 |
| valid | input | 2 | cache行是否有效 |
| byte\_w\_en\_in | input | 4 | 字节写使能 |
| status | input | 3 | 当前cache状态 |
| enable | output | 2 | i/d\_cache使能信号 |
| write | output | 2 | i/d\_cache读写信号 |
| compare | output | 2 | 结合write产生读写模式 |
| i\_byte\_w\_en\_out | output | 4 | i\_cache字节写使能 |
| d\_byte\_w\_en\_out | output | 4 | d\_cache字节写使能 |
| load | output | 2 | 请求载入cache行(cache内部信号) |
| ram\_en\_out | output | 1 | DDR使能 |
| ram\_write\_out | output | 1 | DDR读写(1:写 0:读) |
| d\_cache\_r\_sel | output | 1 | d\_cache行载入端选择(i\_cache或ddr) |
| next\_status | output | 3 | 转移的下一个状态 |

iii)控制器状态转换图



注:状态信号取值见随附表格

(3)i\_cache和d\_cache设计

i)二路组相联cache图

注：上图为一个二路组相联cache，其内部构造为两个128行每行256个字节的直接相联cache并联拼接而成。

ii)信号说明

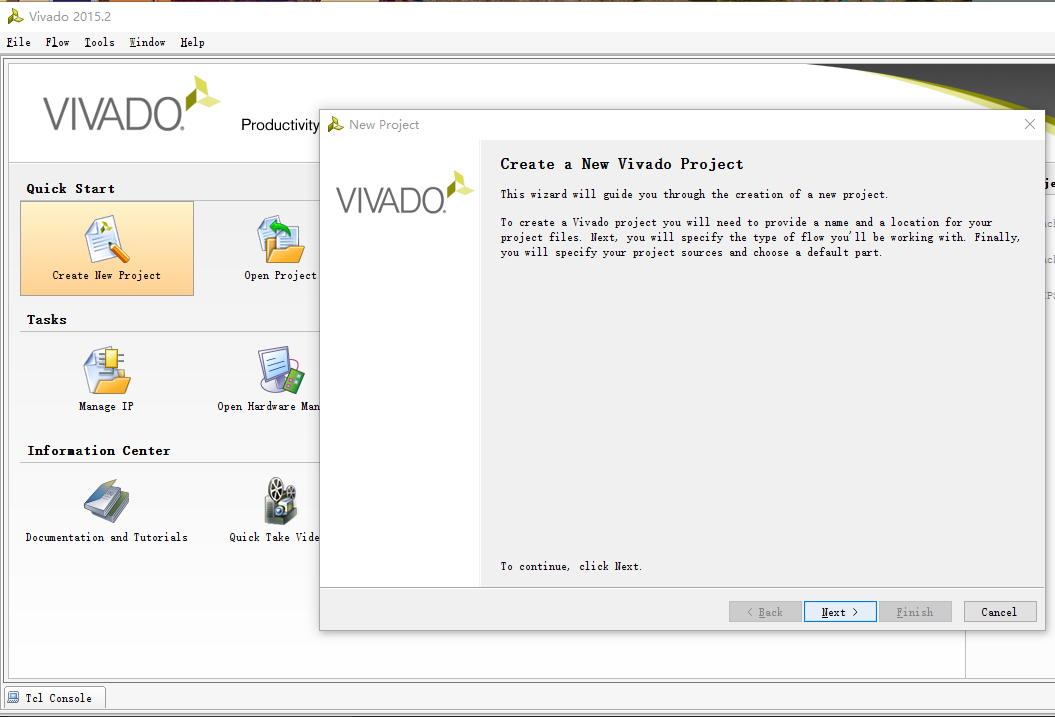
基本信号与顶层模块大致相同，故仅说明几个内部信号

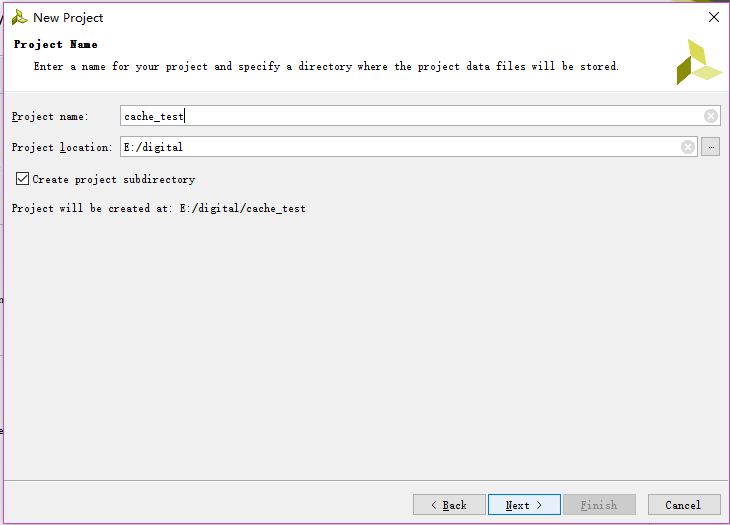
|  |  |  |  |
| --- | --- | --- | --- |
| 信号 | 类型 | 宽度 | 信号说明 |
| tag\_in | input | 20 | tag字段 |
| index | input | 6 | cache组索引 |
| word | input | 3 | cache行内字索引 |
| victim |  | 1 | cache替换行组内行索引 |

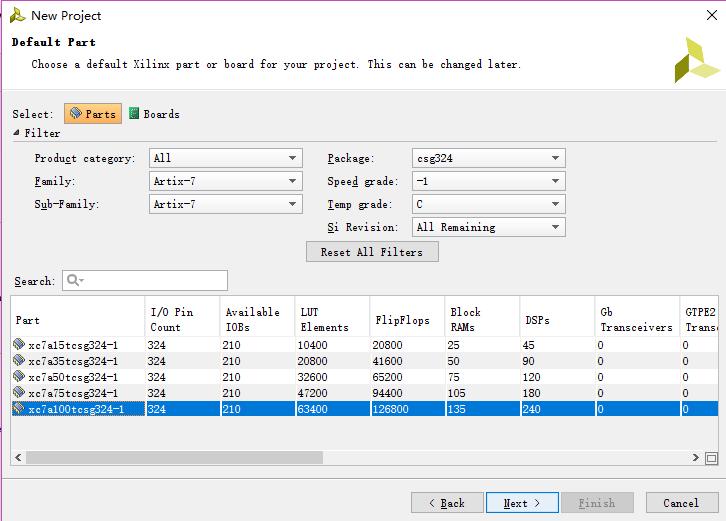
**四、实验步骤**

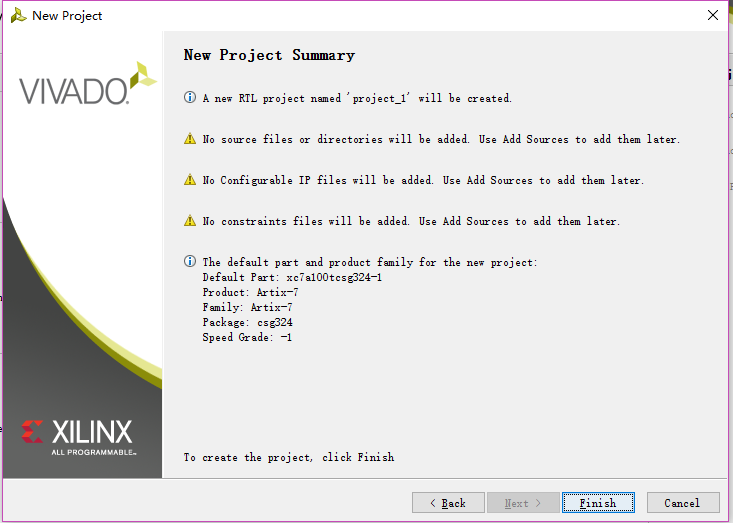
1.cache仿真

(1)新建测试工程

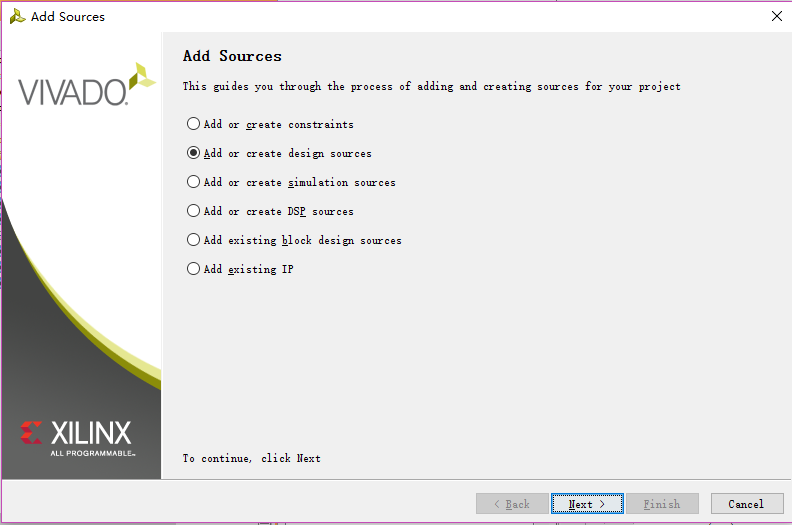


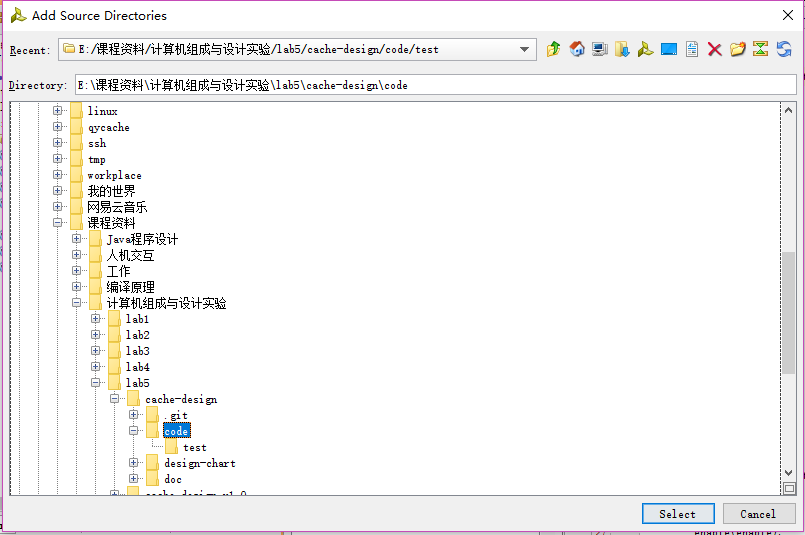






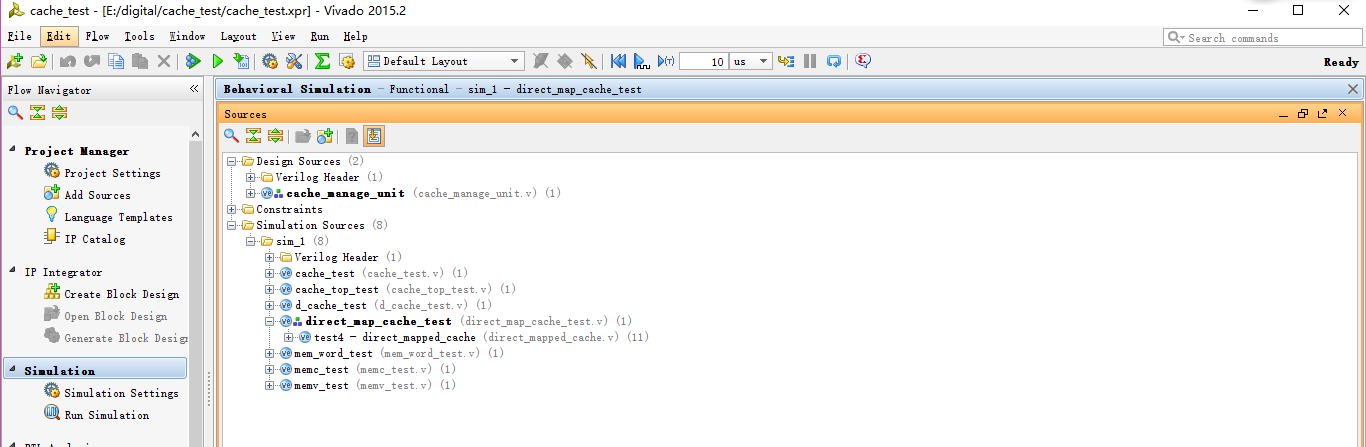
(2)添加源文件及测试文件





点击Select选择添加

最终模块结构如下：



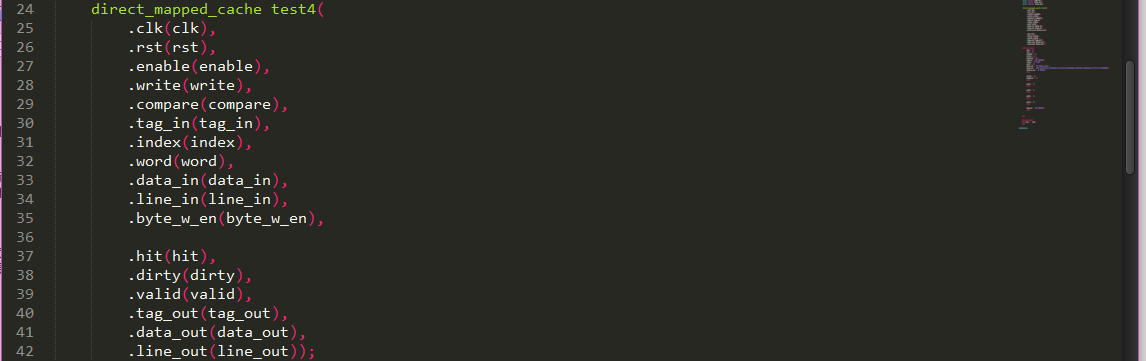
(3)进行仿真

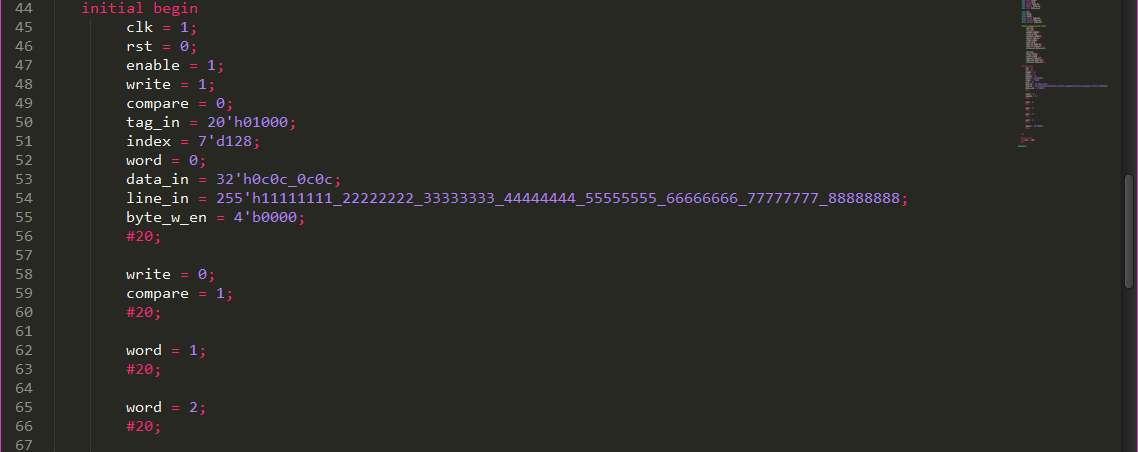
注：测试仿真过程自底向上地进行

i)direct\_mapped\_cache(直接相联cache)

测试代码：

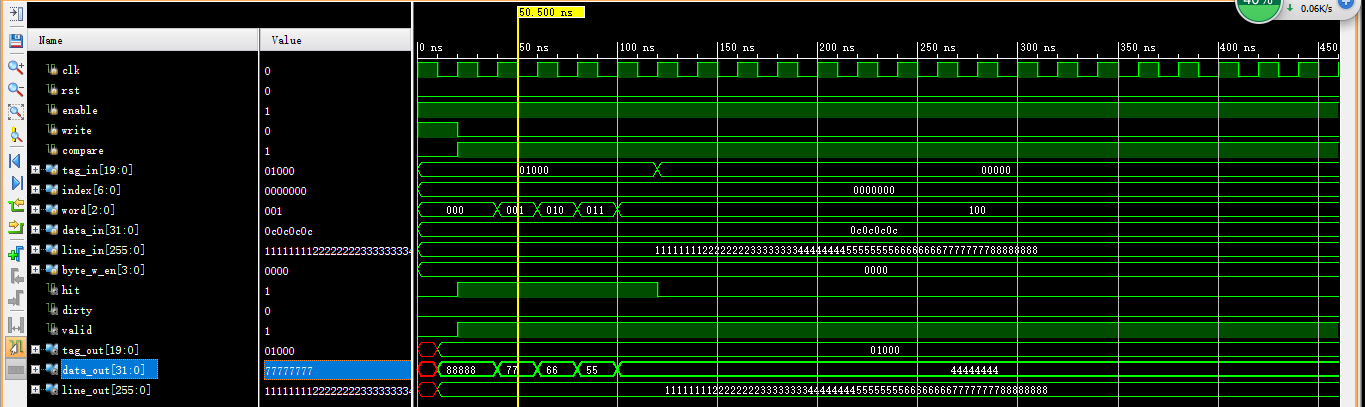








仿真图：



仿真分析：

此段代码主要测试此直接相联cache模块地载入及数据读取功能，先将数据



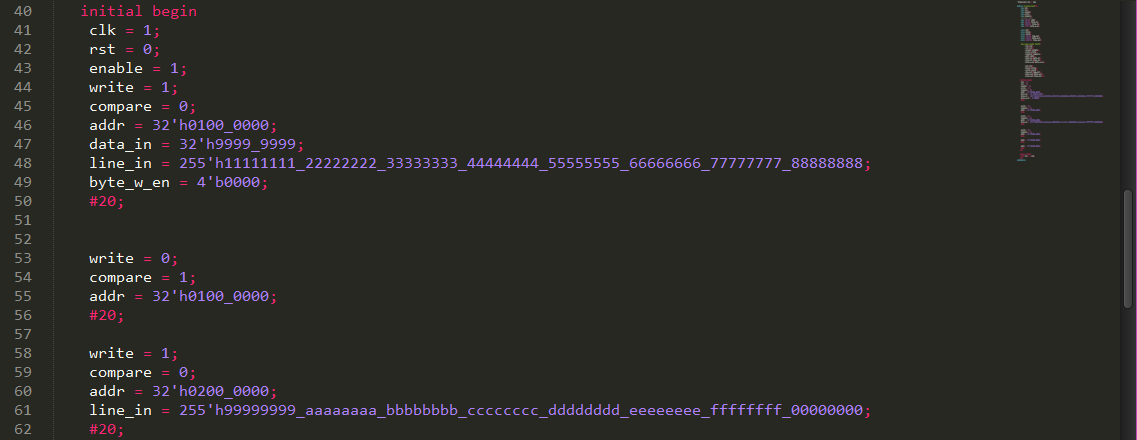
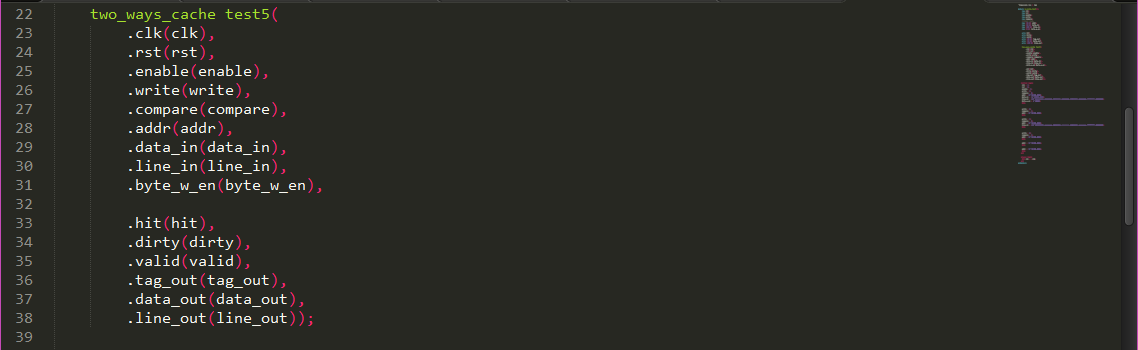
载入某行，并顺次读出各个字，结果显示如下表所示：

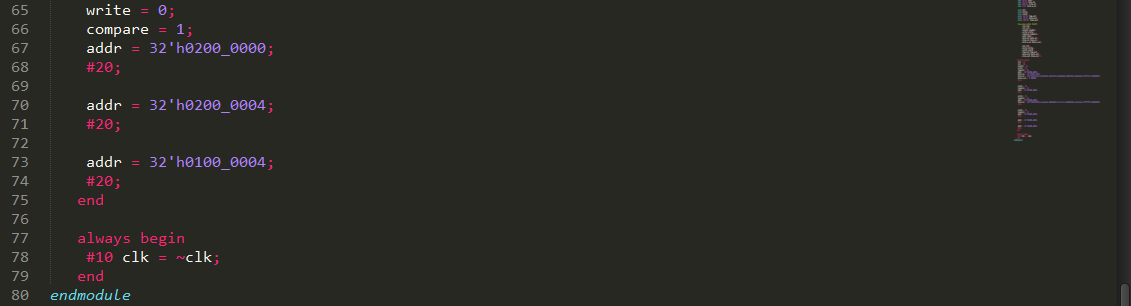
|  |  |
| --- | --- |
| word | data\_out |
| 0 | 88888888 |
| 1 | 77777777 |
| 2 | 66666666 |
| 3 | 55555555 |

ii)two\_ways\_cache(2路组相联cache)仿真

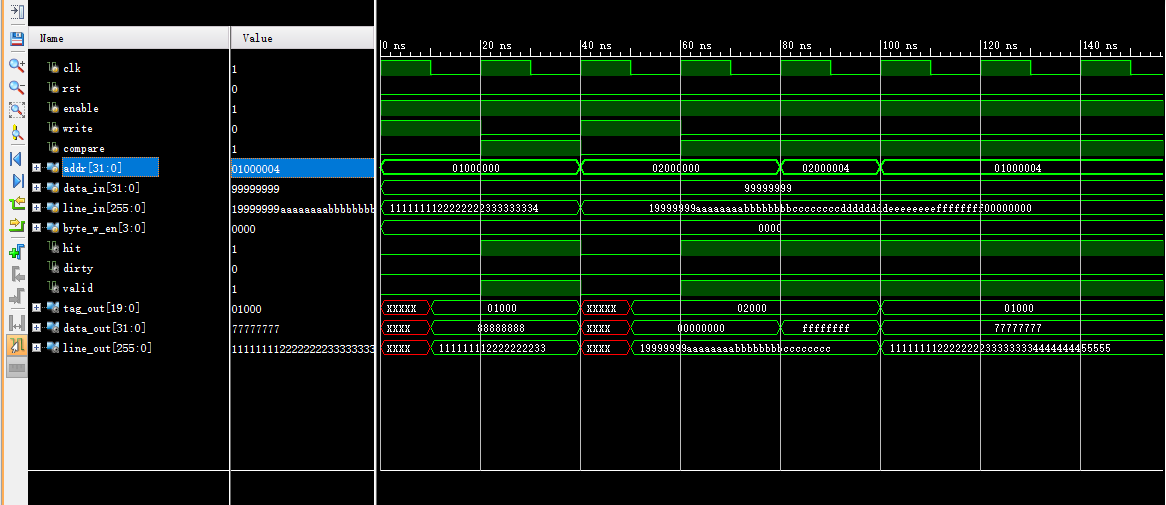
测试代码：







仿真图：



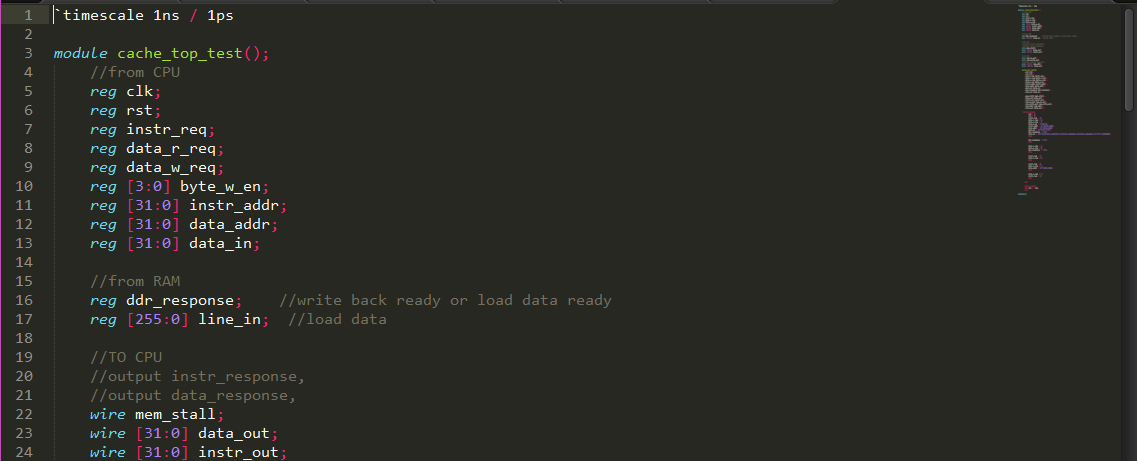
仿真分析：

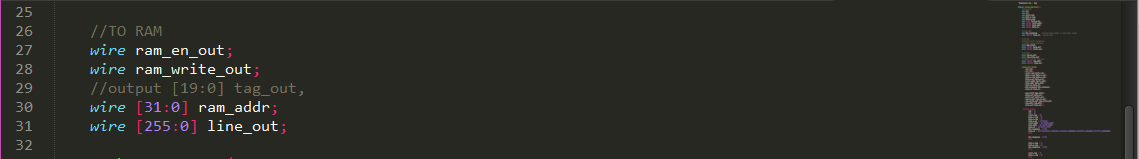
此段代码测试了一个二路组相联cache对同一组的两行进行操作的过程，，构造了两次读cache miss从而引发cache行载入同一组的两行，之后依次读取它们的数据验证cache基本数据读写正确性。结果如下表所示：

|  |  |
| --- | --- |
| addr | data\_out |
| 32'h0200\_0000 | 00000000 |
| 32'h0200\_0004 | ffffffff |
| 32'h0100\_0004 | 77777777 |

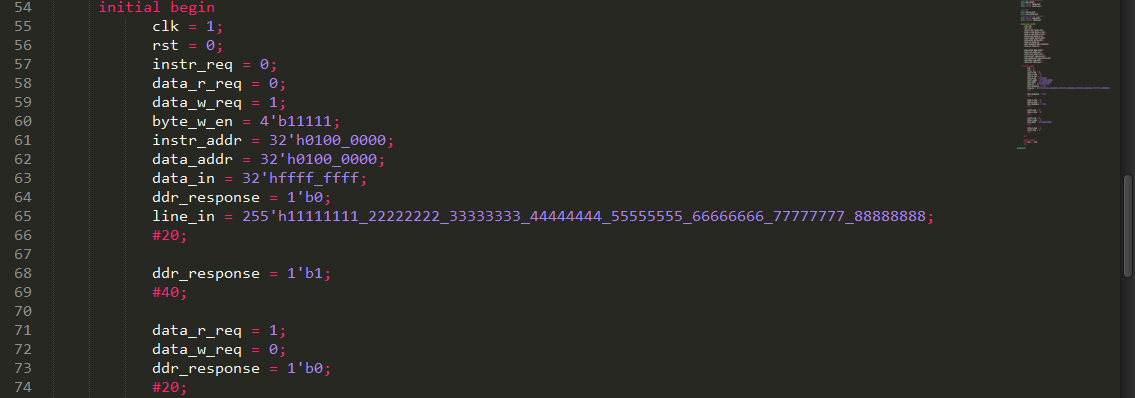
iii)cache\_top(I cache和d cache的顶层)模块

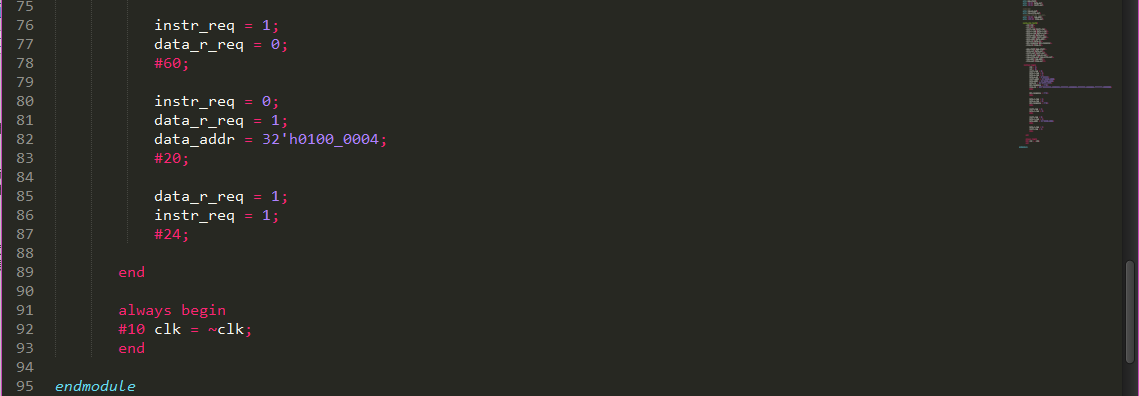
测试代码：



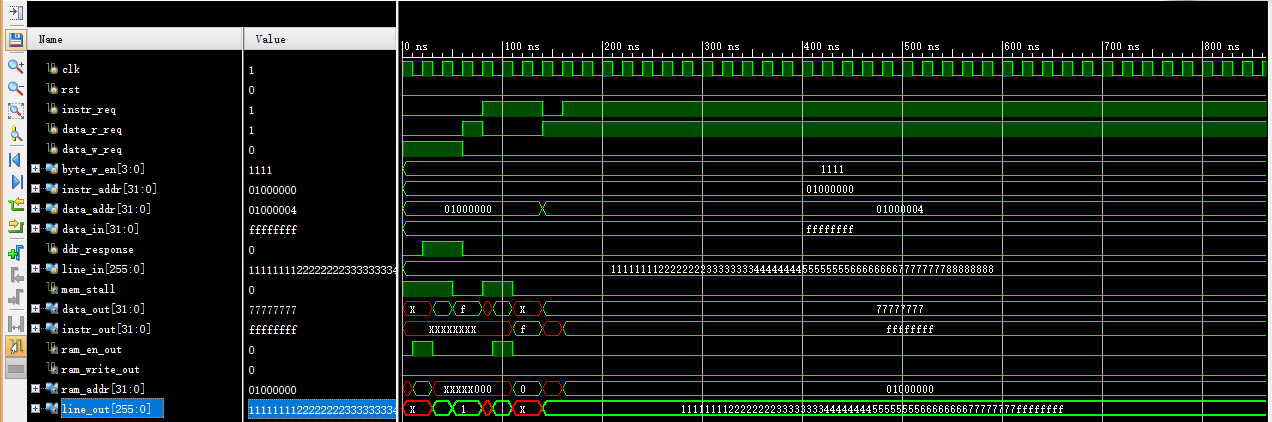








仿真图：



仿真分析：

此段测试代码测试是cache的最顶层模块，构造了一些i\_cache及d\_cache的缺失载入用例，以此测试cache行的替换算法正确性，先在d \_cache中载入一行以0x01000000为首地址的数据，然后再让i\_cache读这一块数据，发现I cache miss后重新从d cache中载入了该行，从而验证了cache一致性问题。

1. 生成二进制文件并用FPGA测试

本次上板测试所使用的方式是将已有的流水线CPU的cache模块替换为自己实现的cache并进行测试，主要测试内容分为两大块，一是基本的读/写数据功能，而是cache行的替换和写回功能，，这两大块各有一个测试文件。

1. 基本读写功能测试

使用一段快速排序的代码进行测试，最终将排序结果打印在屏幕上。

1. cache行的替换和写回功能测试

由于用的是现成的代码，故生成二进制文件和调试时没有出现太大的问题。

**五、实验心得**

本次实验认识和掌握Cache控制器的原理及其设计方法以及掌握Cache控制器的实现方法，代码实现方法。

对于cache的理解更加深入，工作的机制和原理比较清晰。

其中遇到了比较多的问题，逐一上网查询解决办法，并逐个进行尝试，并最终得到了解决，对于个人的信心有极大的提高。

**六、cache代码**

见随附code目录文件